Automatic Classification and Artificial Life Models

Xavier Llorà and Josep M. Garrell

Abstract— *Genetic Artificial Life Environment* **(GALE) is a classification-oriented model based on cooperative agent aggregates spread over a two dimensional board. The model, combining Genetic Algorithms and Artificial Life ideas, solves classification problems descrived by continuous-valued inputs. This paper describes GALE focusing on a key point, resources allocation over the 2D world. The main apportation is that using an accurate resources allocation we can clearly improve convergence speed at the same time that the complexity of the solution is reduced.**

Keywords— **Resources allocation, Agent Hierarchical Aggregates, Artificial Life, Genetic Algorithms and Genetic Programming.**

I. Introduction

GENETIC Algorithms [1], [2] and Genetic Programming [3], [4], [5] are computation models that play a central role in many Artificial Life models [6]. This paper presents one of them, the *Genetic Artificial Life Environment* (GALE) [7], [8]. GALE defines a 2D world where agents cooperate and compete each other looking for survival through adaptation in a certain environment. The main contribution of GALE is that the whole model is oriented to solve machine learning problems -automatic classification problems described by continuous-valued inputs-.

Genetic Algorithms have proved to be a very useful technique in solving automatic classification problems [1], [9], [10]. Likewise, Artificial Life models are able to describe agents' interactions and evolution [1], [11], [12]. GALE mixes up both approaches. This mixture lets us: (1) study the agents' behavior along their evolution, and (2) solve real-world classification problems.

GALE solves classification problems through the emergence of the solution from agents' interactions. These interactions are defined in terms of cooperation and competition. Spontaneous hierarchical adhesion [11] among agents models the cooperation. This adhesion -defined as the aggregation of agents- provides an emergent way of solving the classification problem. Competition is modeled as an arms race for limited space and resources. Cooperation and competition are also the basis for the adaptation of agents to the environment. Agents' adaptation -learning- depends on the evolutionary rules defined in genetic terms, which are based on neighborhood spatial relations.

X. Llorà and J.M. Garrell are with the Intelligent Systems Research Group, Enginyeria i Arquitectura La Salle, Ramon Llull University, Barcelona. E-mail: {xevil,josepmg}@salleURL.edu

A set of classification examples is the input used in GALE for performing supervised learning. The mapping of the input examples over the 2D world of GALE defines resources allocation. This mapping is a key point for the model. We present different criteria for spreading the examples, showing that they can improve the convergence speed of the algorithm at the same time that they reduce the complexity of the solution obtained. In other words, resources allocation can keep agents' interations in a simple fashion, exploiting *building blocks* [1] formation.

In order to complement the description of GALE, we present some results obtained using this model solving a real-world classification problem: the diagnosis of breast cancer. The results are briefly compared to the ones obtained using well-known classification techniques (i.e Neural Networks and Case-Based Reasoning). We also present the behavior of the interactions among agents observed along the GALE runs. Finally we discuss some conclusions and futher work.

II. Genetic Artificial Life Enviroment

As early mentioned, GALE is a classification-oriented world. Therefore, the agents spread over the world are to solve a given classification problem. The meaning of classification in this work is not the process of generating classes or data clusters. This work focuses on supervised learning. The classification problems need to find a way to link an example with its associated class. This means that the set of possible classes is well known for a given problem.

This kind of classification problems can be seen as class prediction. Thus, finding a path between an example and its belonging class is the goal of the evolved agents. In order to reach this goal, the problem provides a set of training examples, each one having its belonging class. Each example feature is continuous-valued.

Definition of GALE holds on four different parts: (1) board topology, (2) agents interactions, (3) world mechanisms and (4) resources allocation. The board topology defines the world appearence and neighborhood relations. Agents and their interations are in charge of solving the classification problem. In other words, the accuracy of GALE depends on their definition. GALE agents evolve following the genetic rules defined by the world mechanisms. Finally, resource allocation is the key for obtaining a good performance

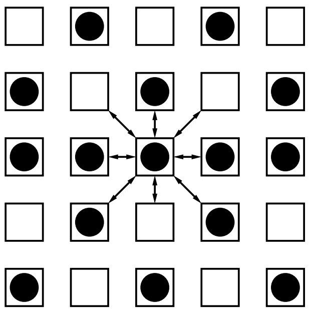


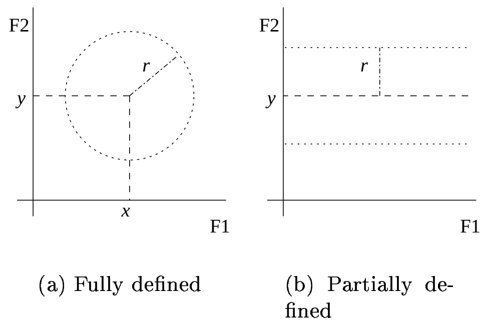
Fig. 1. Board topology. Each cell can be either empty or occupied (painted as a dot)

from the model. It defines how to map the training examples over the world topology.

The board topology is based on a 2D grid. Cellular automata [13] and traditional Artificial Life models [12], [14], [15] inspire this board distribution. The board used is a 2D toroidal mesh. Each cell in the board is connected to its eight surrounding neighbors. This spatial structure constrains the interaction among agents in spacially defined subpopulations. Each cell in the board can be either empty or occupied by an aggregate of agents, shown in figure 1.

III. Agents

Agents are the *building blocks* of the solution to the classification problem. Each agent represents a significant point [16], [17], [18]. This point belongs to the n-dimensional continuous-valued space defined by the features of the classification problem[1]. Each agent is also linked to a class of the classification problem. This class qualifies the classification region defined by the significant point. In other words, all the region share the same classification class.



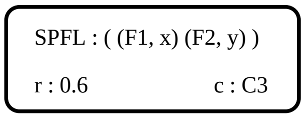
(a) Fully defined (b) Partially defined

Fig. 2. 2D *significant points*

Figure 2.a shows these ideas for a classification problem defined by two continuous-valued features. In this example, the agent represents a point (x,y) in a 2D space defined by the features F1 and F2. The agent also has a limited sight range r that defines its region of influence. All the points in this region share the same class pointed out by the agent. Another characteristic of the agents is that they do not need to fully define significant points. Instead, agents allow representations of partially defined significant points. Figure 2.b presents an example.

An agent codifies the information of its significant point in its genotype. The genotype contains the following information: (1) *Significant Point Feature List* (SPFL), (2) *Threshold* r and *Class* c.

The SPFL contains the value of the features of the significant point described by the agent. The number of features contained in the SPFL is not fixed, so not all the features has to be present at the significant point definition. Threshold r defines the sight range of the agent. Class c represents the linked class associated to the classification region represented by the agent. Figure 3 presents an example of the genotype of an agent.



SPFL : ((F1, x) (F2, y))

r : 0.6 c : C3

Fig. 3. Genotype of an agent

It is important to keep in mind that the main goal of this work wants to solve classification problems described by examples with continous-valued features. An agent can either classify or not an exemple. An agent does not classify an example if it is outside the region defined by the *significant point* codified in its genotype. Otherwise, the agent classifies the example using the class coded in its genotype.

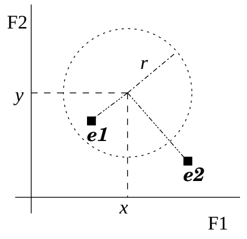


Fig. 4. Example of classification process used by Agents

Figure 4 shows an example of this classification procedure. The example *e2* lays outside the region defined by the agent, therefore *e2* is not classified. Instead, *e1* belongs to the classification region. Thus, the agent classifies *e1* using the class coded in its genotype (c).

The classification of examples uses the distance between the example and the *significant point* of the agent. An agent classifies an example x if:

$$\sqrt{\sum_{i \in SPFL} (SPFL_i - x_i)^2} \leq r \qquad (1)$$

This procedure works correctly in fully defined points -figure 2.a- or partially defined ones -figure 2.b-.

IV. Agent Hierarchical Aggregates

Agent definition has some serious drawbacks. They arise from the limited information coded inside the genotype of each agent. It follows from the matching procedure that an agent all alone does not solve classification problems with more than one class.

[1] n is the number of features of the problem.

Definition also has a limited expressivity for defining region boundaries. These limitations claim an extension of agent capabilities as classifiers. Cooperation and competition among agents show a way for solving these limitations. Evolution plays a main role in the definition of these relations among agents.

Cooperation extends the classification capabilities of agents. Therefore the classification procedure is no longer depending on just one agent. It is the result of the interaction among several agents working together. This means a redefinition of classification regions based on significant points. This redefinition summarizes saying that the classification region of an agent ends where the region of another agent starts. Figure 5 shows the cooperations between three different agents *a1*, *a2* and *a3*. This approach introduces a more expressive way to break the space of features. Choosing the right *significant points* helps us to solve complex classification problems.

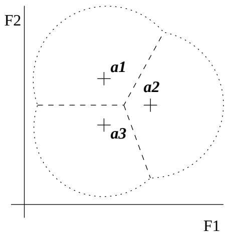


Fig. 5. Three cooperative agents

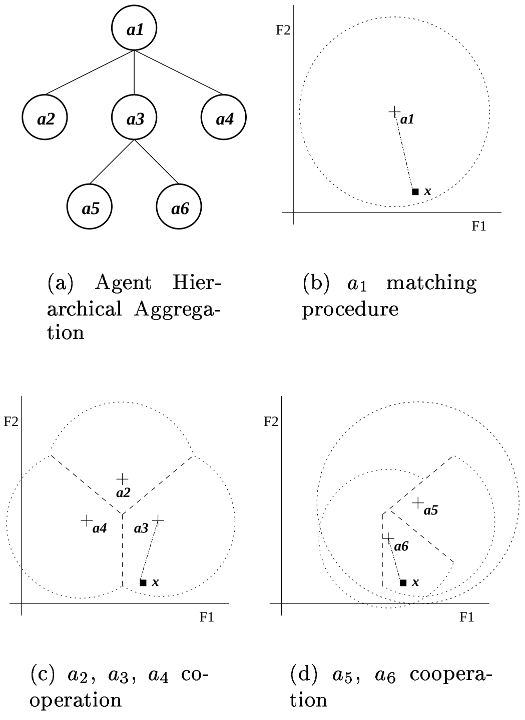
Multiagent structures collect cooperative behavior of agents. These structures stick together useful aggregations obtained through evolution. The structure proposed is a hierarchical aggregation of agents -*Agent Hierarchical Aggregates (AHA)*-. The usage of multiagent structures also needs a hierarchical classification procedure, in order to exploid the cooperative behavior of the structure.

AHAs provide an extended classification procedure. It uses the classification procedure of agents at the same time that introduces *nearest neighbor* policies for defining classification region boundaries. Figure 6 presents how AHAs work out in practice. This example is going to be used for explaining the classification procedure.

Six different agents $\{a_1,\ a_2,\ a_3,\ a_4,\ a_5,\ a_6\}$ constitute the AHA presented in figure 6.a. Agent a_1 is the entry point for classifying the new example x, and it decides to classify the example because x satisfies, as shown in figure 6.b, that:

$$\sqrt{\sum_{i\in a_1^{SPFL_i}} \left(a_1^{SPFL_i} - x_i\right)^2} \leq a_1^r \qquad (2)$$

a_1 knows that x belongs to its classification region because a_1^r threshold is greater or equal to the distance between x and the $SPFL$ of a_1. Once this check is done, a_1 has to decide which is the class associated to x. The resulting class arises from the matching procedure of $\{a2,\ a3,\ a4\}$, figure 6.c. Deciding at which class x belongs is done by choosing the closest agent to



(a) Agent Hierarchical Aggregation

(b) a_1 matching procedure

(c) a_2, a_3, a_4 cooperation

(d) a_5, a_6 cooperation

Fig. 6. Agent hierarchical aggregation for clasification problem solving

x. *Nearest neighbor* policies are the basis of the selection. The selected agent -a_3- is the one that satisfies:

$$\min_{a_i\in vc(a_1,x)} \left(\sqrt{\sum_{j\in a_i^{SPFL}} \left(a_i^{SPFL_j} - x_j\right)^2}\right) \qquad (3)$$

$vc(a^1, x)$ is the set of all agents (children agents of a_1) that has x in their sight range. In other words, it is the set of all child agents that the distance between its $SPFL$ and x is less or equal than its *threshold.* If vc is the empty set then x remains unclassified. Once a_3 is chosen, the classification procedure reaches a_5 and a_6, as figure 6.d shows. The results of this procedure is a_6 -nearest agent- which is in charge of finally defining the class to be linked to x. This class is recovered from a_6 genotype information.

This classification procedure appears as a cooperative endeavour. Cooperation among agents is the basis for the survival of the aggregate since all agents are stuck in a single structure. A detailed explanation of this classification procedure can be found in [7].

V. GALE Mechanisms

Agent hierarchical aggregates emerge as a result of the evolution defined in the GALE model. GALE topology is based on a 2D board -figure 1-. It allows to use a fine-grain parallel evolution based on neighborhood relations. Each cell in the board runs the same algorithm which is in charge of improving the classification accuracy of agent hierarchical aggregates.

Figure 7 presents the three main mechanisms of GALE: (1) survival, (2) merge and (3) split. They mix

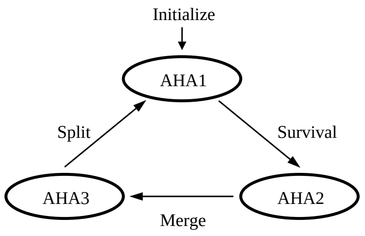


Fig. 7. Cell Cycle

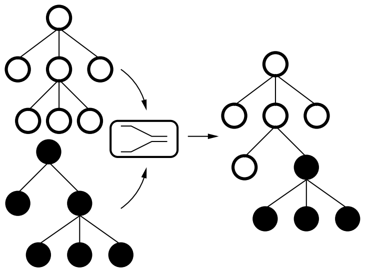


Fig. 8. Aggregates genetic information recombination

up recombination techniques of *genetic algorithms*, *genetic programming* and local neighborhood restrictions inspired by *cellular automata*. A detailed explanation of these mechanisms can be found in [8].

A. Survival

The survival mechanisms implements the selective pressure among cells in the locally defined neighborhoods -figure 1-. The selective pressure for a given cell is defined in terms of AHA *fitness* of the neighborhood and the number of neighbors present. AHA *fitness* is computed as:

$$fitness(AHA) = \left(\frac{c}{t}\right)^2 \tag{4}$$

c is the number of correct classified examples and t is the number of examples contained in the cell. For each cell, the chances of death for the contained AHA varies due to the number of neighbors. Three diferent situations are defined: (1) *solitude* -one or two neighbors-, (2) *steady* -from three up to six neighbors- and (3) *crowd* -more than six neighbors-. These three scenarios define their selective pressure in terms of *death probability* -see [8]-.

B. Merge

The merge mechanism is in charge of the recombination of the genetic material of AHAs. It has two sequential steps: (1) choose the merging neighbor, and (2) recombine the aggregates information. The merging neighbor is randomly selected from the set of available aggregates of cell's neighborhood. Once the merging neighbor is chosen and moved to the cell, AHA information is recombined in terms of genetic operators.

The recombination operator is based on Genetic Programming crossover [3], [4], [5]. Figure 8 shows an example of this recombination process. Recombination produces just one AHA due to the limited room of each cell.

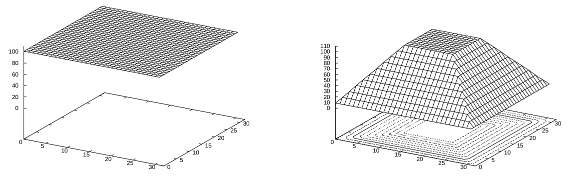
C. Split

The splitting phase is in charge of aggregates autoreplication, introducing genetic diversity through *somatic mutation* [19]. Once the splitted aggregate is obtained, some cell has to contain it. Two rules decide the destination cell of the splited aggregate. These rules can be summarized as: (1) if there are empty neighbor cells, then place the aggregate in the empty cell with more neighbors, and (2) if there are no empty neighbor cells, then replace the neighbor with lowest fitness.

VI. Resouces Allocation

GALE works as a supervised learning model. One key point is how the training examples are going to be spread over the board. In other words, each AHA is evaluated using a set of examples assigned to the cell. Choosing the examples to be placed in each cell is the process we call resouces allocation.

Figure 9 shows two differents ways for resource allocation. The easiest approach, figure 9.a, copies all the training examples in each cell of the board. Doing this, we obtain cells containing the same examples, therefore the same environmental conditions are obtained in each cell.



(a) Uniform (b) Pyramid

Fig. 9. Resource allocation of training examples

The second approach spreads the examples in a nonuniform way. Figure 9.b presents this resources allocation. The distribution spreads the examples using a pyramid shape. Central cells contain all the available examples. The rest contain just a part of the whole set of examples.

The pyramid allocation does not have the same environmental conditions for each cell. Instead, it guarantees that just few changes are introduced in adjacent cells. This means that few examples are removed when moving towards the foot of the pyramid. All the cells with equal number of examples contain the same examples. With this allocation, classification complexity grows when moving throgh cells at the top of the pyramid.

This growing classification complexity encorages *building blocks* formation. Cells at the foot of the mountain define simple environments easy to solve. Due to the parallel definition of GALE board fast exploration of *building blocks* can be achieved. Once discovered they can be exploited in more complex cells through GALE recombination mechanisms. This allocation enables partial solutions, obtained in cells at the foot of the pyramid, reach the cells at the top of the pyramid building a complete solution to the classification problem.

VII. In Practice

GALE testing is done through a real-world classification problem, the breast cancer diagnosis. Using a real-world problem let us test GALE in complex problems. The classification problem deals with the diagnosis of breast cancer in cancerous or not using mamary biopsy images.

Data was provided by the Signal Theory Research Group from our University. The problem can be reduced to the prediction, for each new example, in cancerous or non cancerous. The details of this problem can be found in [16], [20], [21]. Mammary biopsy images are digitally processed and each biopsy image is described by a vector of 24 continous-valued features. Each feature belongs to a bounded continous-valued range. For each processed image, its belonging class -cancerous or non cancerous- is known, provided by the human experts. This vectors are the input data provided to GALE. The data set used contains 1028 continuous-valued vectors.

In order to preliminary test GALE performance, the data set was splitted in two different sets, randomly generated. The first set is the training set used as GALE input. The second one is used in the test phase. The preliminary results presented here uses a 10% of the examples for training and the rest for testing the results obtained from GALE. We also used 90% for training and 10% for testing GALE. For each proportion, several partition were made randomly. For each training set several GALE runs were made.

The tests were designed to prove two hypothesis: (1) GALE is an accurate classification model and (2) resource allocation plays a very important role in spatially defined board models.

TABLE I
Classification performance

	Train	**Test**
Backpropagation Networks	–	82.01%
Case-Based Reasoning	–	80.45%
GALE	99.29%	81.24%

Table I presents the mean classification perfomance of GALE compared to two well-known classification techniques like *Neural Networks* [22] and *Case-Based Reasoning* [23]. The results obtained show that GALE outperforms *Case-Based Reasoning*. They also show that GALE performance is very close to the results obtained using *Neural Networks*.

Figures 10 and 11 prove the hipothesis done for resources allocation. Pyramid shaped allocation has a dual behavior. It clearly improves the convergence speed of GALE, but it also reduce the complexity -number of agents of an AHA- of the solutions evolved by GALE. This results point that resource allocation is actually a key point for the board-based evolution of AHAs.

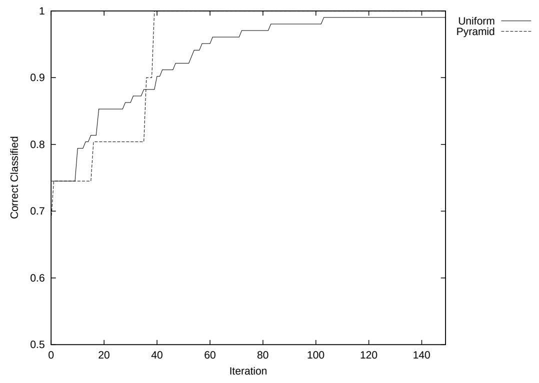
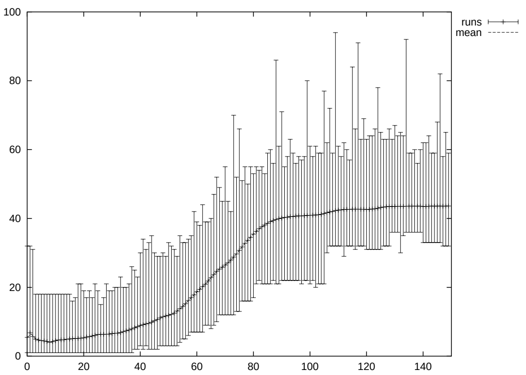
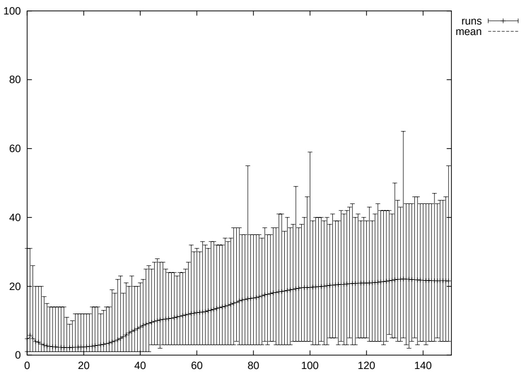


Fig. 10. Comparision between resource allocation approaches



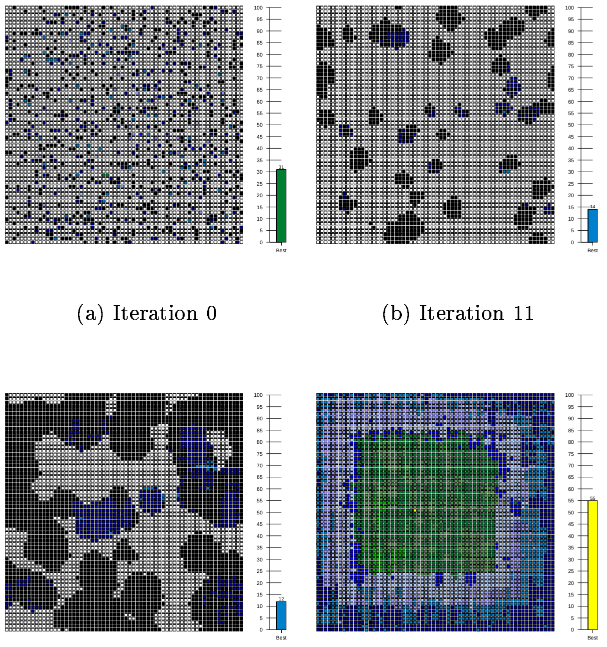
(a) Uniform shaped allocation



(b) Pyramid shaped allocation

Fig. 11. Number of agents in AHAs

GALE also presents another interesting emergent behavior. Evolution, in the initial steps, presents an island growing behavior. Some group of cells remain together. The growing rate of these islands is propotional to the classification accuracy of AHAs in cells. This uncontroled growing leads to a steady state were all cells contain an AHA. This could seem trivial, but the important point is that the complexity -number of agents- of an AHA depends on the resource allocated in each cell. Figure 12 show these facts. Once GALE finds a solution to the classification problem, AHA's complexity remains steady, proportional to the



(a) Iteration 0

(b) Iteration 11

(c) Iteration 22

(d) Iteration 149

Fig. 12. Number of agents of each AHA in the board

number of examples in its cell.

VIII. Conclusions and Futher Work

GALE has presented an 2D world for evolving classification-oriented agent aggregates. This model is based on *Genetic Algorithms*, *Genetic Programming* and *Artificial Life* ideas. Preliminary results point out that it is an efficient model for solving classification problems with continuous-valued inputs. But GALE also shows that resouces allocation in 2D boards is a key point in evolution. Pyramid shaped resource allocation has easely improve the convergence speed of the model at the same time that it reduces the complexity of aggregates.

Some deep work is needed. Different problems must be used for testing GALE. At the same time, the results are compared against the ones obtained with other classification techniques, not only *Neural Networks* and *Cased-Based Reasoning*. Agents classification capabilities and their aggregation structure must be revised and improved. We are currently working with other resource allocation techniques, looking for a better *building blocks* explotation. These resources allocation proposals also deals with recombination mechanisms of GALE.

Acknowledgements

We would like to thank the *Comisionat per a Universitats i Recerca* and the *Direcció General de Recerca* of the *Generalitat de Catalunya* for their suport under grant numbers 1999FI-00719 and D.O.G.C. 30/12/1997. We also would like to thank Enginyeria i Arquitectura La Salle for their support to the Intelligent Systems Research Group.

References

[1] John H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*, MIT Press/ Bradford Books edition, 1992.

[2] David E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Publishing Company, Inc., 1989.

[3] John R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)*, MIT Pres, 1992.

[4] John R. Koza, *Genetic Programming II: Automatic Discovery of Reusable Programs (Complex Adaptive Systems)*, MIT Pres, 1994.

[5] John R. Koza, Forrest H. Bennett III, David Andre, and Martin A. Keane, *Genetic Programming III: Darwinian Invention and Problem Solving (Complex Adaptive Systems)*, MIT Pres, 1999.

[6] Melanie Mitchell and Stephanie Forrest, "Genetic Algorithms and Artificial Life," *Artificial Life: an overview (ed. Christopher G. Langton). MIT Press*, pp. 267–289, 1997.

[7] Xavier Llorà, "*Genetic Artificial Life Environment: solving classification problems using cooperative agents*," Tech. Rep., EALS-2000001, Enginyeria i Arquitectura La Salle, Ramon Llull University, 2000.

[8] Xavier Llorà, "*Genetic Artificial Life Environment: mixing cellular genetic algorithms and artificial life ideas*," Tech. Rep., EALS-2000002, Enginyeria i Arquitectura La Salle, Ramon Llull University, 2000.

[9] Stewart W. Wilson, "Classifier Fitness Based on Accuracy," *Evolutionary Computation*, vol. 3, no. 2, pp. 149–175, 1995.

[10] Stewart W. Wilson, "Get Real! XCS with continous-valued inputs," *Festschrift in Honor of John H. Holland*, pp. 111–121, 1999.

[11] John H. Holland, *Hidden Order: How Adaptation Builds Complexity*, Perseus Books, 1995.

[12] Thomas S. Ray, "An evolutionary approach to synthetic biology: Zen and art of creating life," *ArtificialLife(1/2). Artificial Life: an overview. The MIT Press*, pp. 1–10, 1995.

[13] Stephen Wolfram, *Cellular Automata and Complexity*, Addison-Wesley Publishing Company, Inc., 1994.

[14] John H. Holland, "Escaping Brittleness: The Possibilities of General Purpose Learning Algorithms Applied to Parallel Rule-Based Systems," *Machine Learning: An Artificial Intelligence Approach, Vol. II*, pp. 593–623, 1986.

[15] Christopher G. Langton, "Studying artificial life with cellular automata," *Physica 22D*, pp. 120–149, 1986.

[16] Xavier Llorà and Josep M. Garrell, "GENIFER: A Nearest Neighbour based Classifier System using GA," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO99)*, 1999.

[17] Xavier Llorà, Elisabet Golobardes, and Maria Salamó, "Diagnosis of microcalsification using case-based and genetic algorithms," in *Proceedings of Engineering of Intelligent Systems (EIS2000)*, 2000.

[18] Xavier Llorà and Josep M. Garrell, "Evolving Agent Hierarchical Aggregates using Cellular Genetic Algorithms," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, 2000.

[19] S. Forrest and A. Perelson, "Genetic Algorithms and the Inmune System," in *Parallel Problem Solving by Nature, Berlin. Springer-Verlag*, 1991.

[20] E. Martínez, C. Vos, and et al., "Morphological analysis of mammary biopsy images," in *Proceedings of the IEEE International Conference on Image Processing (ICIP'96)*, 1996, pp. 943–947.

[21] Josep M. Garrell, Elisabet Golobardes, Ester Bernadó, and Xavier Llorà, "Automatic diagnosis with genetic algorithms and case-based reasoning," in *Journal of Art. Int. in Engineering, 13(4):367-372. Elsevier Science Ltd.*, 1998.

[22] James A. Freeman and David M. Skapura, *Neural Networks*, Addison Wesley, 1993.

[23] C.K. Riesbeck and R.C. Schanck, *Inside Case-Based Reasoning*, Lawrence Erlbaunm Associates, Hillsdale, 1989.